RDF Data Descriptions

Georg Lausen[1] and Michael Schmidt[2]

[1] University of Freiburg, Institute for Computer Science, 79110 Freiburg, Germany
lausen@informatik.uni-freiburg.de

[2] Kaiserstraße 86, 66133 Scheidt, Germany
m.schmidt00@gmail.com

Abstract. Linked Open Data (LOD) sources on the Web are increasingly becoming more popular. RDF constraints can be used to characterize the RDF graphs being provided by such sources. For applications that process data retrieved from several of such RDF graphs it becomes interesting to analyze the relationships of the different sets of constraints associated with the sources providing the RDF graphs. In this short paper we discuss how the constraints from different sources can be aggregated to a set of constraints characterizing the union of the RDF graphs under consideration. For expressing constraints we use Datalog+/-.

The recent RDF Validation Workshop [9] states a gap between the current standards offering and the industry needs for validation of RDF data. As a possible solution, in continuation of our previous work [7], we have developed a constraint language RDD (RDF Data Descriptions) [8], that captures a broad range of constraints including keys, cardinalities, subclass, and subproperty restrictions, making it easy to implement RDD checkers and clearing the way for semantic query optimization.

The intention of an RDD is similar to Stardog ICV [1], where constraints are stated using OWL and considered relative to a certain inference machinery whose type may range from no inferencing, RDFS- to OWL-inferencing. In contrast, RDD is a language using a compact special-purpose syntax designed for only expressing constraints independent of a specific inference machinery. This makes RDD in particular applicable for RDF under ground semantics, which is a common scenario in the Linked Data context.

While in [8] we considered a restricted scenario where a single RDD defines the constraints given in a single RDF graph, in this short paper we suggest to broaden the view to a set of RDF graphs, each described by its own RDD defining a set of associated constraints. The major difficulties of such a scenario arise as the information represented by the graphs may overlap in the sense that certain resources may be described in more than one graph. To accomplish such situations the notion of a

context has been coined [6]. While in this paper RDF in different context is discussed with respect to *information* aggregation, our concern is aggregation of *constraints*, which has not been studied before, to the best of our knowledge.

Let us consider RDF graphs G_a and G_b and corresponding RDDs RDDa and RDDb, respectively. Both graphs are assumed to be consistent, i.e. all the constraints in the respective RDD are fulfilled. The main question we are interested in is how the union of the graphs $G_a \cup G_b$ is related to the union of the respective sets of constraints Σ_a and Σ_b. As G_a and G_b may contain triples referring to subjects with the same URI, in general it will hold $G_a \cup G_b$ does not fulfill all constraints in $\Sigma_a \cup \Sigma_b$. For example, whenever a certain predicate p is defined to be single-valued in RDDa and RDDb, then two corresponding triples (s, p, o_1) and (s, p, o_2) may appear in the union $G_a \cup G_b$ of both graphs, so that the constraint is not guaranteed to hold in $G_a \cup G_b$. As solution for such cases we propose *aggregation* of constraints what in our example would mean that the single-valued constraint is replaced by a constraint restricting the number of occurrences of values to 2. In general, given RDDs Σ_a and Σ_b, we are interested to construct an RDD $\Gamma(\Sigma_a, \Sigma_b)$ such that for any RDF graphs G_a and G_b, where $G_a \models \Sigma_a$ and $G_b \models \Sigma_b$, we have $G_a \models \Gamma(\Sigma_a, \Sigma_b)$, $G_b \models \Gamma(\Sigma_a, \Sigma_b)$ and $G_a \cup G_b \models \Gamma(\Sigma_a, \Sigma_b)$. The task of deriving $\Gamma(\Sigma_a, \Sigma_b)$ is called *constraint aggregation*.

Recently, Cortés-Calabuig and Paredaens [4] have presented a constraint language for RDF equipped with deductive rules for equality and tuple generating dependencies. However, as can be seen from the following example, their constraint language is not general enough to be used for an RDD. For these reasons we have chosen the framework of Datalog+/- [2], which offers the needed expressiveness.

In Figure 1 we exhibit two RDDs describing RDF graphs representing employees and projects. To demonstrate constraint aggregation let us consider predicate `reportsTo`, which is defined via a path-constraint in RDDa. In RDDb for `reportsTo` it is defined that each employee must report to exactly two objects. These constraints are written in Datalog+/- as follows (predicate names are abbreviated):

$$G(\$s, rT, \$o) \rightarrow \exists \$o_1 (G(\$s, wF, \$o_1), G(\$o_1, aT, \$o))$$
$$G(\$s, rT, \$o_1), G(\$s, rT, \$o_2), G(\$s, rT, \$o_3) \rightarrow \$o_1 = \$o_2 \vee \$o_1 = \$o_3 \vee \$o_2 = \$o_3$$
$$G(\$s, type, E) \rightarrow \exists \$o_1, o_2 (G(\$s, rT, \$o_1), G(\$s, rT, \$o_2), \$o_1 \neq \$o_2)$$

Moreover, RDDa defines `worksFor` and `assignedTo` as total functions. Therefore it can be inferred that `reportsTo` is a partial function, even though this is not stated in RDDa. Using this additional information,

```
PREFIX ex: <http://www.example.com#>
CWA CLASS ex:Employee {
  KEY rdfs:label : LITERAL
  PARTIAL ex:employedBy : RESOURCE
  MAX(2) ex:prevEmployedBy : RESOURCE
  TOTAL ex:worksFor, RANGE (ex:Project)
  PATH(ex:worksFor/ex:assignedTo)
     ex:reportsTo, RANGE(ex:Consortium) }

CWA CLASS ex:Project {
   TOTAL ex:assignedTo,
     RANGE(ex:Consortium)  }
```

```
PREFIX ex: <http://www.example.com#>
CWA CLASS ex:Employee {
  KEY rdfs:label : LITERAL
  PARTIAL ex:employedBy : RESOURCE
  ex:prevEmployedBy : RESOURCE,
    SUBPROPERTY employedBy
  MIN(2), MAX(2) ex:reportsTo,
    RANGE(ex:Association) }
```

Fig. 1. RDDa (left) and RDDb (right) describing employees in different contexts. See [8] for a detailed explanation of the used concepts.

by constraint aggregation we get a min(0)- and max(3)-constraint for predicate `reportsTo`. Note that without the inferred constraint, predicate `reportsTo` has to be considered to be unrestricted and therefore only the trivial constraint $max(\infty)$ can be derived by constraint aggregation. Formally, inferring constraints in the Datalog+/- framework can be done based on the chase procedure [5]. We are currently investigating termination and complexity of the corresponding constraint implication problem. However, constraint aggregation, as proposed in this paper, by itself is independent from the concrete constraint language considered. For example, in [3] for a wide number of ontology languages termination and efficiency of the required chase procedure is demonstrated.

References

1. Stardog. http://http://Stardog.com/.
2. Andrea Calì, Georg Gottlob, and Thomas Lukasiewicz. A general datalog-based framework for tractable query answering over ontologies. *J. Web Sem.*, 14:57–83, 2012.
3. Andrea Calì, Georg Gottlob, and Andreas Pieris. Towards more expressive ontology languages: The query answering problem. *Artif. Intell.*, 193:87–128, 2012.
4. Alvaro Cortés-Calabuig and Jan Paredaens. Semantics of constraints in rdfs. In *AMW*, pages 75–90, 2012.
5. Alin Deutsch and Alan Nash. Chase. In *Encyclopedia of Database Systems*, pages 323–327. 2009.
6. R. Guha, R. Mccool, and R. Fikes. Contexts for the semantic web. In *ISWC*, pages 32–46. Springer, 2004.
7. Georg Lausen, Michael Meier, and Michael Schmidt. SPARQLing Constraints for RDF. In *EDBT*, pages 499–509, 2008.
8. Michael Schmidt and Georg Lausen. Pleasantly consuming linked data with rdf data descriptions. In *COLD*, volume 1034 of *CEUR Workshop Proc.*, 2013.
9. W3C. Rdf validation workshop, practical assurances for quality rdf data. http://www.w3.org/2012/12/rdf-val/, 2013.